WWW.SCIAM-MAGAZINE.NAROD.RU

Занимательный компьютер

О том, когда игру можно считать работой, и о неопасных терминальных болезнях

А.К. ДЬЮДНИ

В ИССЛЕДОВАТЕЛЬСКОМ центре фирмы Bell Laboratories в Мюррей-Хилл (шт. Нью-Джерси) невозможно провести грань между работой и игрой. Время от времени безудержная фантазия и удивительная изобретательность сотрудников центра порождают игры, имеющие весьма серьезное значение. Показательные в этом смысле примеры дают терминал типа Блит (Blit) и развлекательная программа КРАБЫ (CRABS), которая иногда мешает работать с системой Блит.

Разработанная четыре года назад Р. Пайком и Б. Локанти, система Блит представляла собой предварительную версию нового терминала DMD 5620 фирмы Teletype Corporation. Система фирмы Teletype и ее прообраз Блит представляют собой терминалы, работающие в многопрограммном режиме.

Когда прошлой осенью я посетил исследовательский центр в Мюррей-Хилл, Пайк предложил мне посмотреть терминал DMD 5620 в действии. На терминалах этого типа (которые я ниже буду просто называть терминалами Блит) экран может быть поделен на отдельные окна, иногда частично перекрывающиеся. Каждое окно, по существу, представляет собой самостоятельный миниатюрный экран, полностью отданный в распоряжение какой-либо одной программы. Программы могут выполняться встроенным в терминал микропроцессором или большой центральной ЭВМ, к которой подключен терминал. При этом незанятые области экрана покрываются точечным узором, как бы окрашивающим их в серый цвет.

На экране терминала, показанного мне Пайком, три окна были заняты тремя одновременно выполняющимися программами. Одно окно принадлежало находящейся в стадии отладки экспериментальной программе редактирования текстов. Во втором окне программа-редактор показывала текст первой программы. Третье окно было отведено программе отладки, при помощи которой осуществлялась проверка экспериментальной программы редактирования текстов. Все три программы выполнялись встроенным в терминал Блит микропроцессором. Пайк демонстрировал передо мной необычайные возможности этой многопрограммной системы, как вдруг в верхней части экрана появилось несколько крабовидных символов, разбросанных вдоль границы окна. «Крабы» двигались по экрану, как бы поедая изображение в окне. Заметив мое удивление, Пайк объяснил: «Это крабы. Чтобы вас позабавить, мы только что загрузили программу КРАБЫ из центральной машины».

Центральная машина, к которой был подключен терминал Блит, работала под управлением операционной системы UNIX. Одна из наиболее популярных операционных систем за всю историю вычислительной техники, система UNIX, была создана К. Томпсоном, сотрудником фирмы Bell Laboratories, разделившим премию Тьюринга за 1983 г. со своим коллегой из центра в Мюррей-Хилл Д. Ритчи.

Завороженный, я наблюдал, как крабы (их было около тридцати) съели линию, ограничивающую одно из окон, и начали поедать текст внутри. «Как их можно остановить?» — спросил я. «Никак», — ответил Пайк. «Да, но ведь вы их каким-то образом останавливаете?»

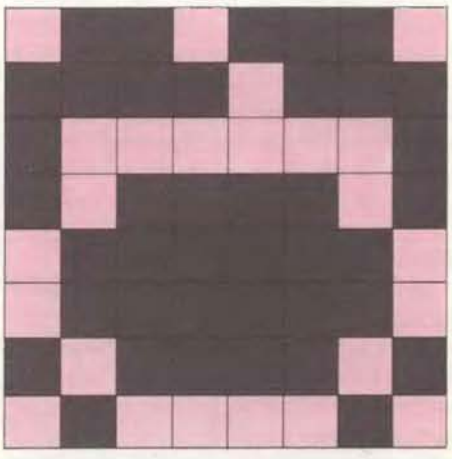
Программа КРАБЫ, написанная Л. Карделли и М. Манэссе в 1982 г., — это преднамеренное нарушение соглашений, принятых в многопрограммной системе, с которой работает терминал Блит. Дело в том, что каждая программа, имеющая свое окно на экране, должна быть независимой от программ из других окон и защищенной от них. Когда терминалы системы Блит были расставлены по помещениям лабораторий, Карделли и Манэссе не смогли удержаться от соблазна нарушить эти соглашения.

К тому времени, как Пайк поведал мне эту историю, все окна на экране были уже изрядно поедены. Они напоминали страницы недавно найденной средневековой рукописи. Текст в окнах стал настолько неразборчивым, что Пайк уже не мог объяснить мне, как протекает процесс выполнения соответствующих программ.

Пайк рассказал, что название этого оригинального терминала произошло от имени графической процедуры низкого уровня bitblt, отличающейся удивительно широким спектром применения. Имя bitblt в свою очередь является сокращением от слов bit — boundary block transfer (букв. перенос блоков с битовой границей. — *Перев.*), обозначающих процедуру, которая управляет перемещением информации по памяти терминала. Точнее говоря, bitblt переносит содержимое одной прямоугольной области памяти в другую. В ходе этого переноса над содержимым области могут производиться некоторые простые логические операции.

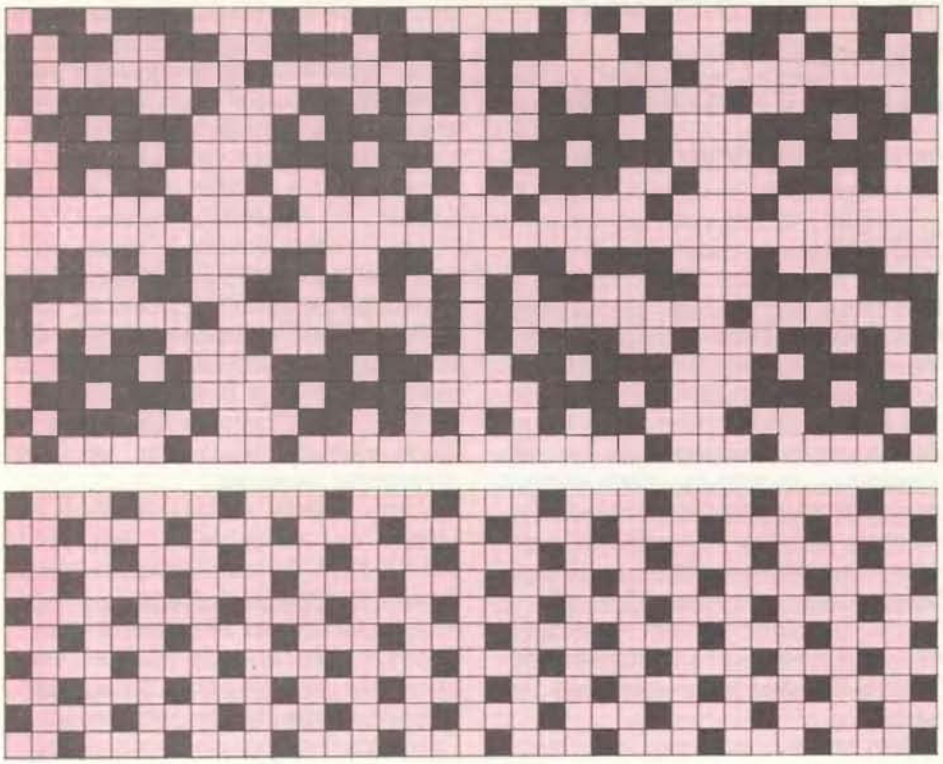
Часть памяти терминала Блит состоит из подсистемы емкостью 100 000 байт, целиком отведенной для экрана. В процессе непрерывного сканирования каждый бит памяти преобразуется в светлую или темную точку на экране дисплея. При этом одномерный массив памяти преобразуется в двумерное изображение содержимого массива. Поскольку область памяти и ее изображение имеют различные размерности, точки, принадлежащие прямоугольной области на экране, оказываются разбросанными по памяти в виде отдельных блоков ячеек с последовательными адресами, так что каждый блок соответствует одной горизонтальной линии (полоске) на экране, из которых состоит прямоугольник. Совокупность точек-ячеек во всех блоках образует прямоугольное множество. Процедура bitblt позволяет выполнять определенные операции над такими прямоугольными множествами.

Для успешной работы в многопрограммной среде терминал Блит требует также значительного объема основной, «неэкранной» памяти, необходимой программам, диалог с которыми проходит с использованием экрана терминала. Например, в тер-



Изображение краба на квадратной решетке

WWW.SCIAM-MAGAZINE.NAROD.RU



Битовые карты краба со светлыми точками, нанесенными на них в соответствии с восемью возможными способами наложения краба на серое поле

минале DMD 5620 фирмы Teletype для этой цели отводится 900 000 байт. Здесь также важную роль играют прямоугольные множества. К процедуре bitblt часто происходят обращения, когда надо преобразовать прямоугольное множество основной памяти в прямоугольное множество памяти экрана. Пусть, например, в некотором прямоугольном множестве основной памяти содержится буква А. Всякий раз, когда на экране должна появиться буква А как часть текста, выводимого программой, происходит обращение к процедуре bitblt, переносящей это прямоугольное множество в экранную память.

Аппаратное сканирование памяти — это принципиальная конструктивная особенность, присущая графическим терминалам с описанным выше буфером памяти для экрана, или так называемым «буфером-фреймом». Подобные дисплеи получают все более широкое распространение и уже применяются в персональных компьютерах. Свидетельство тому — дисплей персонального компьютера фирмы MacIntosh.

В традиционных системах с графическими дисплеями обращения к памяти происходят по более сложным алгоритмам. Представление рисунка на экране осуществляется, например, при помощи списка точек, линий и других графических элементов. Для того чтобы высветить рисунок на экране, необходимо просмотреть список. При этом простые рисунки можно высветить быстро, потому что их списки короче, однако для изображения сложных рисунков может потребоваться слишком много времени.

Конфигурация битов в какой-нибудь прямоугольной области памяти называется битовой картой. Каждая битовая карта представляет на экране определенное изображение: типографский знак, простой рисунок-символ или еще какой-нибудь графический элемент. В программе КРАБЫ, например, используются некоторые битовые карты из основной памяти, при помощи которых на экране строятся крошечные ракообразные различной формы. Краба представляет следующая цепочка из нулей и единиц (единицы соответствуют темным, а нули — светлым точкам на экране):

01101110. . .11110111. . .10000001
. . .10111101. . .01111110. . .
01111110. . .10111101. . .01000010

Читатели, желающие промоделировать bitblt, могут нарисовать на листе бумаги квадратную решетку размерами 8×8. Тогда приведенная выше цепочка будет представлять прямоугольную область в основной памяти, а квадратная решетка — часть экранной памяти. Скопируйте цепочку на решетку, просканировав побитно ту и другую. Теперь можно воспроизвести изображение на решетке, закрасив черным цветом каждую ячейку, содержащую единицу, и оставив светлыми ячейки, содержащие нули. При этом мы получим изображение краба, занимающего небольшую часть экрана (см. рисунок на с. 73).

Выше я уже отмечал, что любая часть экрана, не охваченная никаким окном, закрашивается точечным узором в серый цвет. Когда крабы напали на экран терминала Пайка, они сначала ползали по серой области. Натолкнувшись на окно, они начали его поедать. При этом Пайк шутливо заметил, что «серый цвет — это дорожки для крабов, а своих дорожек они не поедают». На самом деле крабы ничего не поедают, а скорее асфальтируют. Когда программа КРАБЫ двигает свое детище, она проверяет поле, находящееся непосредственно на пути краба; если оно не серое, то программа окрашивает его в серый цвет. Затем она перемещает краба в только что обработанную область. Обе операции осуществляются при помощи процедуры bitblt.

В простейшем варианте работы процедура bitblt заменяет каждый бит d на битовой карте, являющейся результатом, соответствующим битом s исходной битовой карты. Кратко эту операцию замещения можно обозначить формулой

$$d \leftarrow s.$$

Любую область экрана можно сделать серой, воспользовавшись этим вариантом процедуры bitblt. Поле, находящееся непосредственно на пути краба, можно рассматривать как битовую карту-результат, а битовую карту в основной памяти, содержащую шаблон серого цвета, можно считать исходной битовой картой.

Затем программа КРАБЫ перемещает краба. Она стирает изображение краба в том месте, где оно в данный момент находится, и высвечивает его на новом месте. Чтобы выполнить соответствующие операции, используется более сложный вариант процедуры bitblt, который можно условно обозначить следующим образом:

$$d \leftarrow s + d.$$

Здесь каждый бит поля результата замещается его комбинацией с соответствующим битом источника. Эта комбинация представляет собой логическую операцию исключающего ИЛИ, которую мы впредь будем обозначать как XOR или знаком «+ ». В результате применения операции XOR бит d будет равен 1 тогда и только тогда, когда либо s, либо d (но не оба одновременно) имеют значение, равное 1. Операцией XOR очень удобно пользоваться как при стирании старого изображения краба, так и при построении нового. К сожалению, в

WWW.SCIAM-MAGAZINE.NAROD.RU

этом варианте процедура bitblt не может воспользоваться стандартным изображением краба, которое мы получили выше при помощи карандаша и бумаги.

Посмотрим, например, что будет, если операцию XOR применить к стандартной битовой карте краба и точечному шаблону серого цвета. Когда операция XOR применяется к точке серого поля (1) и какой-либо точке тела краба (также 1), результатом будет светлая точка (1 + 1 = 0). Если, однако, данная точка тела краба является светлой, то результатом будет темная точка (0 + 1 = 1). Этим объясняется странный вид восьми специальных битовых карт краба, которыми пользуется программа КРАБЫ при перемещении своих подопечных (см. рисунок на с. 74). Каждая из этих битовых карт представляет краба со специальным подбором светлых точек на его теле. Существует восемь возможных способов поместить краба на серое поле, и каждому из этих способов соответствует своя битовая карта краба.

Программа перемещает краба, сначала стирая его, а затем высвечивая на новом месте. Чтобы стереть краба, процедура bitblt отображает при помощи операции XOR битовую карту краба с соответствующим подбором светлых точек на светящееся в данный момент изображение краба. В результате изображение краба исчезает и на его месте остается серое поле. Чтобы высветить новое изображение краба, процедура bitblt опять пользуется специально подобранной битовой картой краба, которую она отображает посредством операции XOR на серое поле. Новое изображение краба ничем не отличается от старого, поскольку светлые точки на его битовой карте точно соответствуют серому полю, на которое краб перемещается.

Алгоритм, управляющий поведением крабов, можно разбить на пять шагов:

1. Стереть изображение краба в том месте, которое оно в данный момент занимает.

2. Определить новое место.

3. Если новое место не является серым полем, то сделать его серым, изменить данные о положении краба и выбрать случайным образом новое значение скорости передвижения.

4. Высветить краба на новом месте.

5. Изменить случайным образом скорость краба. Вернуться к шагу 1.

Крабы, как и полагается, движутся боком. Поэтому новое место всегда находится сбоку от краба, по ту или другую сторону от него. Новое место определяется текущими значениями направления и скорости движения краба. Обе эти величины подвергаются на шаге 5 случайным изменениям. Возможны несколько направлений движения, а его скорость варьируется в пределах от 1 до 7 пикселов (ячеек на экране) на одно перемещение.

Во время нашего разговора с Пайком крабы сначала появились в верхней части экрана. Затем, совершая хаотические перемещения по серым полям, они стали медленно спускаться вниз. Встретив окно, они прогрызли его границу и проникли внутрь. Было хорошо видно, как они прогрызают окно, потому что на изображении стали появляться крошечные серые пятна. Внешне это выглядело очень эффектно: крабы как будто пожирали верхний слой бумаги, под которым проступало серое волокно.

В значительной степени эта иллюзия объяснялась шагом 3 алгоритма программы КРАБЫ, поскольку на этом шаге краб уходит с только что проеденного им места так, словно он посидел на нем какое-то мгновение, пережевывая проглоченный кусок, прежде чем отправиться дальше в поисках новой порции. Благодаря случайному характеру движения крабов окно становится «дырявым» сразу на большой поверхности. Если бы не эта особенность их движения, то эффект, производимый крабами, скорее напоминал бы процесс выкапывания нор. Благодаря случайному характеру движения крабов человек за терминалом не сразу понимает, что происходит, и его реакция оказывается замедленной.

Пока мы наблюдали за тем, как крабы проедали окна на экране терминала, Пайк рассказал мне о том, как по-разному реагировали его коллеги, впервые столкнувшись с нападением крошечных ракообразных. Часто такое нападение устраивалось исподтишка. Когда ничего не подозревающий сотрудник отходил на минутку от терминала, кто-нибудь из шутников запускал на его терминале программу КРАБЫ.

Поскольку терминал Блит снабжается обычно устройством «мышь», управляющим перемещением курсора на экране, сбитый с толку сотрудник иногда лихорадочно пытался противостоять крабам при помощи этого устройства. Однако все подобные попытки оказывались тщетными, так как крабы с таким же аппетитом набрасывались на курсор, как и на текст. При этом «съеденный» курсор через некоторое время вновь появлялся на экране. И, хотя наступление крабов можно таким образом несколько замедлить, остановить их невозможно и изображение в окнах продолжает разрушаться.

Иногда программист, чей терминал подвергся нападению крабов, пытался накрыть их окном. Эта мера также несколько замедляла продвижение крабов, однако они постепенно прогрызали вокруг себя все большее поле и в конце концов вырывались из окна.

Я снова спросил Пайка, каким образом можно остановить крабов. На этот раз он смилостивился: «Выключите терминал и включите его снова». Такая мера оказывается действенной, однако при этом пользователю приходится заново подключаться к центральной ЭВМ.

Карделли и Манэссе, авторы программы КРАБЫ, изложили историю происхождения программы в неофициальной докладной записке. Конструкция и правила пользования терминалом Блит произвели на них очень сильное впечатление, и они решили проанализировать возможности нарушения этих правил. Они обратились к Пайку, который дал им текст интересовавших их фрагментов программного обеспечения системы. Поразмыслив, двое программистов пришли к выводу, что некоторые возможности нарушения правил, принятых в конструкции терминала, могут оказаться полезными. В этом смысле примером служит программа ЛИНЗА, которая увеличивает заданную часть экрана. (На рисунках на с. 76-77 видно окно программы ЛИНЗА, увеличивающее текст, пораженный крабами.) ЛИНЗА представляет собой нарушение правил, потому что программа, имеющая одно окно, не должна иметь прямого доступа к содержимому другого окна.

Вначале Карделли и Манэссе намеревались имитировать видеоигру QIX, но затем вместо этого написали программу МИКРОБЫ (MEASLES), в результате выполнения которой по серым участкам экрана между окнами начинали плавать маленькие шарики. Первые же испытания показали, что микробов можно было заблокировать, накрыв их окном. Авторы сочувствовали своим созданиям: «Внезапно бедные маленькие микробы оказались запертыми, им было некуда двигаться, нигде вокруг не было спасительного серого поля. Они застыли, полные ужаса, и были похоронены под окном».

Первая реакция авторов на эту неудачу заключалась в том, чтобы заставить пойманных микробов периодически вспыхивать, досаждая тем самым человеку, который поймал их. Новая программа получила название ЗЛЫЕ МИКРОБЫ.

Однако даже «злые микробы» были ничто по сравнению со следующим замыслом, пришедшим в голову неутомимым авторам: «А почему бы

WWW.SCIAM-MAGAZINE.NARODRU

микробам не прогрызть себе путь и выйти таким образом из окна?» Соответствующая программа получила имя ГОЛОДНЫЕ МИКРОБЫ. Карделли пишет: «У новой версии был уже совсем другой характер. Она уже не была просто остроумной, она внушала трепет».

Кто-то, наблюдая за работой программы на экране, заметил, что микробы напоминают чем-то крошечных ракообразных. Карделли и Манэссе реализовали изображение краба, и таким образом родилась программа КРАБЫ. Следом за ней появились и другие программы подобного рода, в том числе такие, которые оставляли на экране отпечатки следов птиц и зверей.

Сотрудники, ставшие жертвой этих шуток, принимали ответные меры. Была написана программа ДАВИ КРАБОВ, которая просматривала память терминала в поисках программы КРАБЫ. Найденные программы уничтожались, а на экране крабы превращались в маленькие пузырьки. К сожалению, очередная модификация в программном обеспечении терминала сделала программу ДАВИ КРАБОВ неэффективной.

В лабораториях исследовательского центра в Мюррей-Хилл есть и другие занимательные программы. Меня познакомили с одной из таких программ, написанной Р. Хардином. Она превращает произвольный текст на английском языке в цепочки фонем, которые затем перегруппировываются, снова превращаются в слова английского языка и выводятся в виде нового текста. Результирующий текст (после некоторого редактирования человеком) выглядит как стихотворение слегка сумасшедшего поэта:

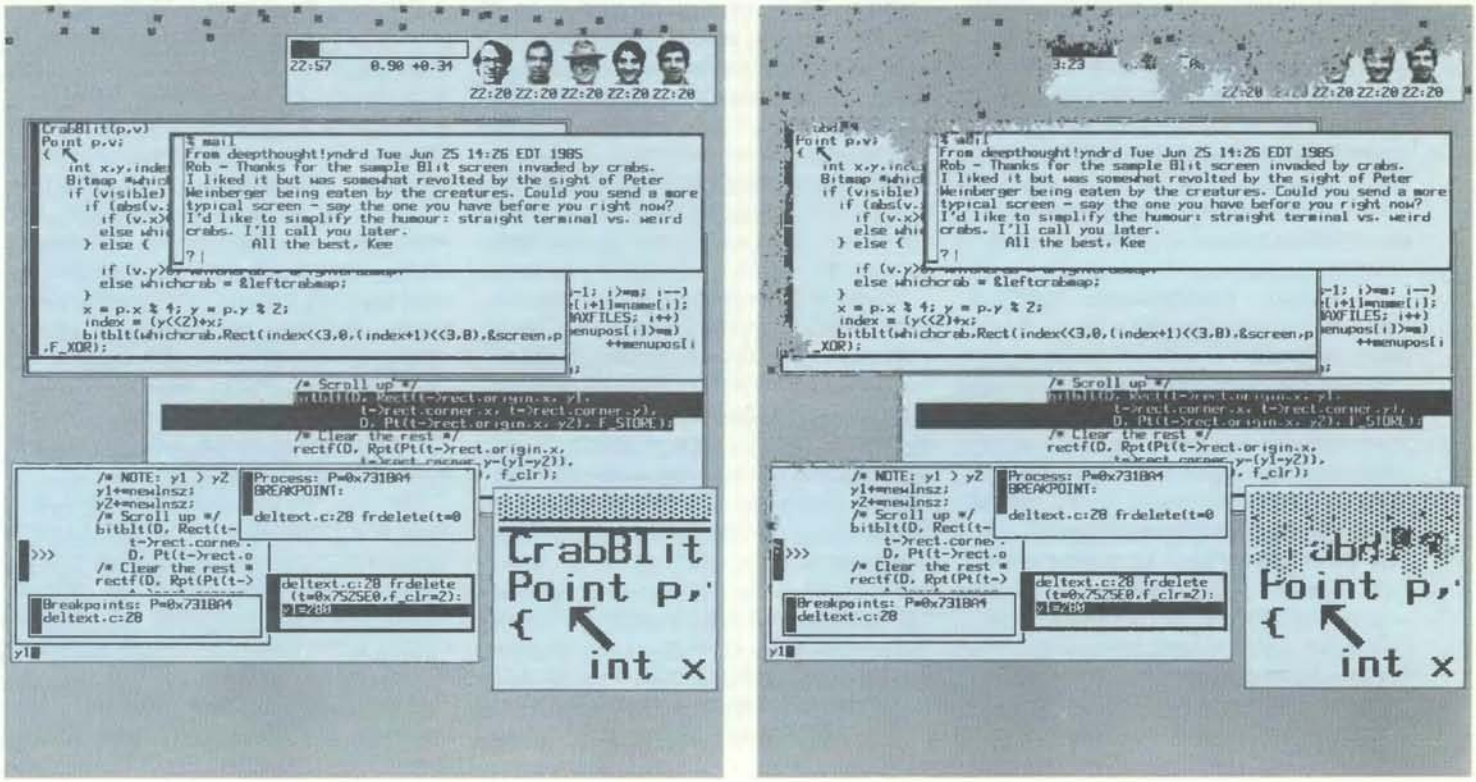
Better buy perfect Topeka beagle buffers
Topeka beagle buffers sputter fiber beaks
Effeter fiber beaks abetted feeble puppies
Worst Topeka beagle puppies feature viper-pique.

Несколько более серьезная работа была проведена при создании программы для игры в шахматы. В 1980 г. чемпионат мира по шахматам среди ЭВМ был выигран системой Belle, представлявшей собой комплекс из специализированной ЭВМ и программы, написанной в лабораториях центра в Мюррей-Хилл. Авторы программы — К. Томпсон и Д. Кондон. (Я думаю посвятить этой и другим современным программам для игры в шахматы одну из своих следующих статей.)

Компьютер, играющий в шахматы, созданный, казалось бы, для развлечения, возвращает нас к разговору о соотношении между работой и игрой. Профессор А. Перлис, преподающий информатику в Йельском университете, в своей недавно опубликованной книге «Структура и интерпретация программ для ЭВМ» (см. «Библиографию» на с. 84) высказывает мнение, получившее довольно широкое распространение среди специалистов по вычислительной технике и информатике.

«Я думаю, очень важно, чтобы в научной работе всегда оставалось место для шутки и развлечения. Когда вычислительная техника и программирование только зарождались, в нашей работе было много веселого и забавного.» Однако, полагает Перлис, сейчас для этого остается все меньше места, потому что исследователям предъявляются серьезные требования, заставляющие их все в большей степени сосредоточивать внимание на «гайках» и «винтиках», из которых состоят изобретаемые ими системы. Но такими вещами следует скорее заниматься тем, кто внедряет научные достижения в производственные системы. «В последнее время мы стали чувствовать себя так, будто бы мы действительно несем ответственность за успешное, безошибочное функционирование этих систем. Мне кажется, это неправильно. Я думаю, наша роль заключается в том, чтобы расширять их возможности, открывать для них новые направления и поддерживать в лабораториях веселое настроение.»

Продуктивность научных исследований в вычислительной технике, как в серьезных, так и в шуточных разработках, объясняется прежде всего свободной творческой атмосферой, в которой новые идеи, по-видимому, порождаются наполовину людьми, а наполовину самими машинами. Можно даже сказать, что без этого веселого настроения не было бы и быстрого



Экран терминала, показывающий вывод из пяти программ, подвергшихся атаке крабов

WWW.SCIAM-MAGAZINE.NAROD.RU

научного прогресса.

Перед тем как закончить статью, я не могу не вернуться еще раз к терминалу Блит и его процедуре bitblt. Пайк предложил две интересные задачки, которые демонстрируют эффективность и гибкость процедуры bitblt. Предположим, что читатель располагает двумя версиями процедуры bitblt, описанными выше. Одна из них замещает битовую карту в поле результата исходной битовой картой. Другая выполняет операцию XOR.

Задача 1. Очистить прямоугольник. Не прибегая к специальной, заполненной единицами битовой карте в основной памяти терминала, воспользуйтесь процедурой bitblt либо в версии замещения, либо в версии с операцией XOR, чтобы очистить заданный прямоугольник на экране.

Задача 2. Повернуть изображение. Квадратное изображение размером $n \times n$ занимает верхний левый угол экрана Блит размером $2n \times 2n$. Воспользуйтесь процедурой bitblt, чтобы повернуть изображение на 90°. Считайте, что вне изображения экран чист.

В первой задаче не разрешается делать никаких предположений о содержимом внешней по отношению к заданному прямоугольнику части экрана. Какой должна быть в этой ситуации битовая карта, которая могла бы послужить исходной для процедуры bitblt?

В обеих задачах воздействие процедуры bitblt на заданный прямоугольник (не важно, какой формы) считается одной операцией. Первая задача решается однократным обращением к процедуре bitblt, вторая требует $3n + 1$ таких обращений. Сможет ли кто-нибудь найти более быстрое решение?

В АВГУСТОВСКОМ номере журнала я предложил читателям найти изящное доказательство утверждения о том, что при помощи приема Дж. Венмакера можно действительно отыскать самый длинный путь на «дереве», сделанном из шнурков. Я говорил, что для определения самого длинного пути достаточно двух операций подвешивания. Сначала нужно взять дерево за любой узел и подвесить его. Затем нужно взять дерево за самый нижний узел и снова подвесить его. Самый длинный путь проходит от узла, за который было подвешено дерево во второй раз, до узла, который теперь оказался самым нижним.

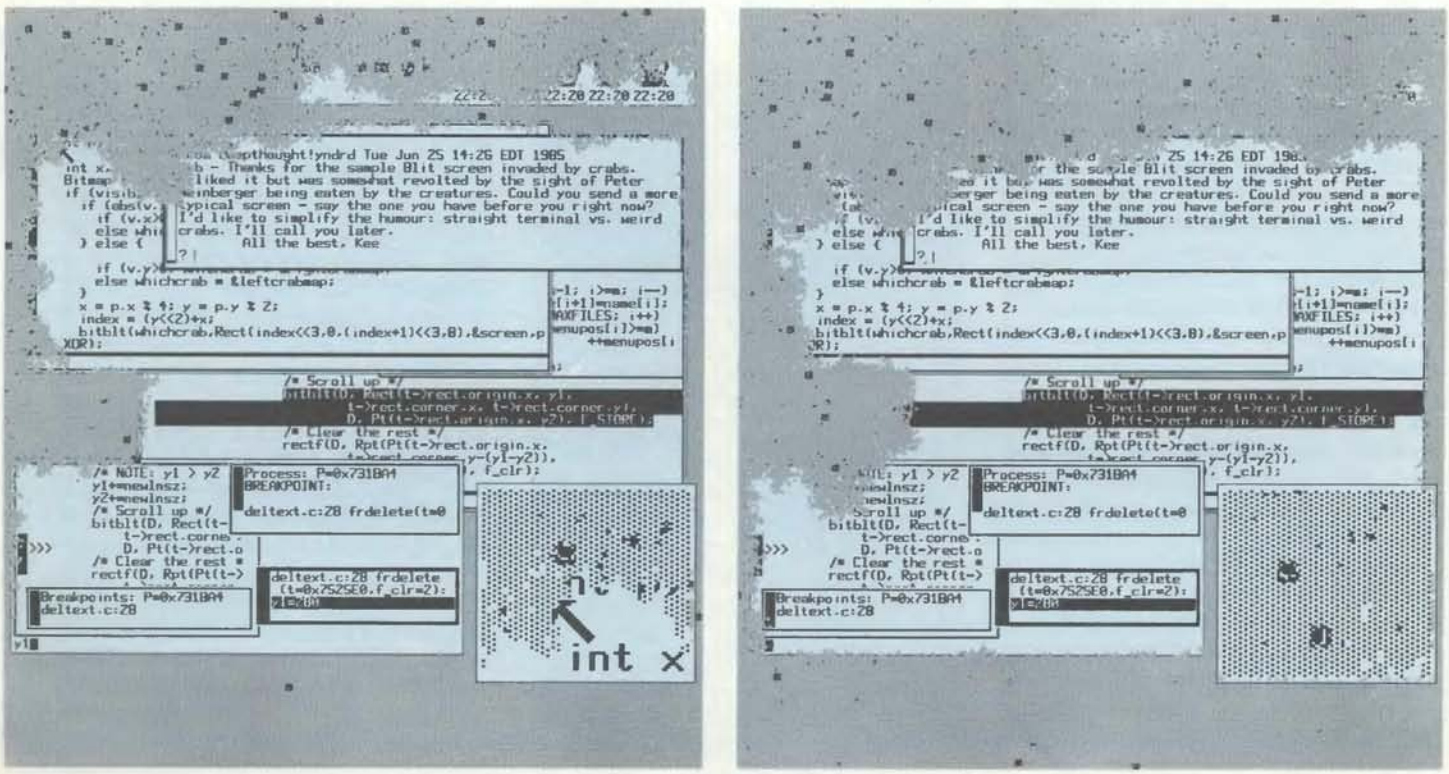
Выше мы неявно предполагали, что дерево имеет единственный самый длинный путь. На самом деле у дерева может быть несколько различных путей одинаковой длины, которая превышает длины всех остальных путей. Следовательно, на обоих шагах процедуры у нас есть выбор между несколькими узлами и в такой ситуации можно выбрать один из таких узлов произвольно.

Мне прислали значительно больше доказательств, чем я ожидал. Премию за самое изящное доказательство можно было бы присудить почти на равных основаниях следующим читателям: Дж. Бартольди III из Атланты (шт. Джорджия), Н. Дершовицу из Эрбаны (шт. Иллинойс), У. Липпу из Милфорда (шт. Коннектикут), Д. Миду из Дейвиса (шт. Калифорния), и Дж. ван-Бентему из Гронингена (Голландия). Нетрудно догадаться, что все эти лица, за исключением Липпа, так или иначе связаны с университетами. Это соответственно Технологический институт шт. Джорджия, Иллинойсский университет в Эрбана-Шампейн, Калифорнийский университет в Дейвисе и Гронингенский государственный университет.

Под «изящным доказательством» я подразумеваю сочетание краткости и ясности. В рамках нашей рубрики это означает также минимальное количество обозначений, уравнений и неравенств. С учетом всех этих требований победителем можно считать Бартольди. Вот его доказательство:

«Пусть H — точка подвеса, а L — нижняя точка. Утверждение: L должна быть одним из концов самого длинного пути.

Доказательство: Пусть A — другой конец самого длинного пути и $A \neq L$. Путь максимальной длины, одним из концов которого является точка A, состоит из участка, ведущего вверх, и участка, ведущего вниз. Однако первый участок не превосходит расстояния от A до H, а второй участок не превосходит расстояния от H до L. Таким образом, путь от A до L представляет собой путь максимальной длины.



Пользователь может отбить подобное нападение, только выключив терминал и повторно подключившись к системе

WWW.SCIAM-MAGAZINE.NAROD.RU

Издательство **МИР** *предлагает:*

Ленг К.
АСТРОФИЗИЧЕСКИЕ ФОРМУЛЫ.
Руководство для физиков и астрономов

В двух книгах
Перевод с английского

Книга 1 включает разделы, посвященные непрерывному излучению и монохроматическому излучению в спектральных линиях. Приведены законы излучения, уравнения распространения электромагнитных волн, уравнения переноса; атомные параметры и спектры, характеристики линий, условия их возбуждения и т.д.

Книга 2 посвящена процессам в газах (столкновениям, переносу, газодинамике, гравитационной неустойчивости), ядерной астрофизике и частицам высоких энергий. Сюда же включены материалы по земному сфероиду, небесным координатам, массам и движению небесных тел и основным положениям теории относительности.

Справочник служит ценным пособием для физиков, астрономов, астрофизиков, математиков, механиков, которые найдут в нем сведения и литературные ссылки, до того разбросанные по разным источникам. Справочник может быть полезен также студентам старших курсов.

«Мир», 1978, 55 л.
Цена 5 р. 50 к. за комплект.

Заказы на эту книгу направляйте по адресу:
103050, Москва, Петровка, 15,
магазин № 8 «Техническая книга».

Если теперь подвесить дерево за точку L, то по аналогичным соображениям точка A окажется самой нижней».

Надо сказать, что приведенное доказательство скорее интуитивное, чем строгое. Тем не менее оно выглядит убедительным, потому что мы представляем себе мысленно, как участки самого длинного пути, проходящего через A, которые ведут соответственно вверх и вниз, висят рядом с путем от H к L. Строгое доказательство потребовало бы значительно больше места.

Первое устройство из тех, которые мы обсуждали в августовском выпуске журнала, состояло из доски, нескольких гвоздей, резиновых колец и длинного гладкого стержня. Это устройство позволяет найти прямую, наилучшим образом аппроксимирующую несколько экспериментальных точек на графике. Я говорил, что в доску нужно вбить гвозди, представляющие экспериментальные точки, затем накинуть резиновое кольцо на каждый гвоздь и на стержень, после чего следует отпустить стержень, который, подрожав, быстро приходит в равновесие.

Я утверждал, что равновесное положение стержня минимизирует полную потенциальную энергию системы (что правильно) и что, следовательно, сумма расстояний от точек до стержня, т.е. сумма отрезков, перпендикулярных стержню, должна быть минимальной (что неправильно). Я имел в виду формулу для силы, которая обусловлена растяжением резинки и пропорциональна ее удлинению. Однако, как заметили многие читатели, энергия, запасенная каждой резинкой, пропорциональна квадрату ее удлинения. Чувство стыда за эту оплошность было, однако, более чем скомпенсировано. Я понял, что устройство М. Хоули гораздо ближе к цели — вычислению по методу наименьших квадратов, чем я ожидал. Остается одно важное различие в том, что следует считать расстоянием. Для вычисления по методу наименьших квадратов нужно, чтобы резинки были натянуты вертикально, а не перпендикулярно стержню (если только стержень не занимает горизонтального положения).

Несколько читателей заметили, что резиновые кольца должны быть идеальными в том смысле, чтобы у них не было провисающих участков. Этому требованию можно удовлетворить следующими способами. Ч. Диллингем из Нью-Йорка (шт. Нью-Йорк), заменяет гвозди на цилиндры. Длина такого цилиндра равна длине нерастянутой резинки. Каждая резинка прикрепляется к доске внутри цилиндра. Резинка удлиняется, только когда ее вытягивают из цилиндра и накидывают на стержень. Р. Файньюкейн из Уотербьюри (шт. Вермонт), решает эту проблему, пропуская каждую резинку через отверстие на доске, соответствующее экспериментальной точке.

Гидравлическое устройство для решения кубического уравнения не вызвало критических отзывов, за исключением замеченной несколькими читателями путаницы. Дело в том, что члены третьей и второй степеней уравнения представляются соответственно конусом и параболоидом, а не наоборот. В этом можно убедиться, проведя несложные вычисления.

Я думаю, что пройдет довольно значительное время, прежде чем мы сможем вернуться в наших статьях к аналоговым устройствам. Позвольте мне, так сказать, на прощание представить вам картонное устройство для подсчета площади земельного участка, присланное А. Лейзаром из Тинека (шт. Нью-Джерси).

Лейзар пользуется картонной коробкой, которую можно найти у любого продовольственного магазина, и портативными весами, чтобы проверять точность вычисления «площади земельного участка с нерегулярной границей». Он наносит контуры участка на лист картона и вырезает модель ножницами. Затем он взвешивает полученную модель и квадратик того же картона, отвечающий в принимаемом им масштабе площади 1 акр. Поделив первый вес на второй, он получает площадь участка.

Со всех концов света продолжает поступать так много результатов расчетов по программе PINT, что я вынужден опубликовать новую среднюю величину. При подсчете были учтены все оценки числа π, вычисленные по 1000 выстрелам и полученные с тех пор, как было подсчитано значение, приведенное в сентябрьском номере журнала. В течение двухмесячного периода накопления данных я наблюдал, как среднее значение постепенно уменьшалось, достигнув в какой-то момент 3,129. Положение несколько улучшилось, когда прибыла почта из Европы и Азии, и среднее значение снова начало подниматься. Хочу выразить особую признательность многочисленным участникам, которые читают издание нашего журнала на немецком языке (Spectrum der Wissenschaft). Благодаря им округленное среднее значение на сегодняшний день оказалось правильным с точностью до двух знаков после запятой. Всего было получено 126 писем, удовлетворяющих поставленным требованиям, и теперь среднее значение числа π равно 3,137.